A Scalable Adaptation of the OLSR Protocol for Large Clustered Mobile Ad hoc Networks

Lucile Canourgues, Jerome Lephay, Laurent Soyer and Andre-Luc Beylot

Abstract Intense research activity is focused on solutions for the routing function in Mobile Ad hoc Networks (MANETs). The most recent efforts emphasize the scalability in large-scale networks, which arises as one of the major challenges to solve in the near future. This paper presents our solution for scalability of the Optimized Link State Routing (OLSR) protocol, the well known proactive routing protocol for ad hoc networks. Based on the Internet experience, which suggests that hierarchy is an interesting solution, our protocol assumes that nodes are grouped into clusters thanks to a clustering algorithm. We offer theoretical and simulation analyses of the control overhead generated by our approach, and we compare it to the Fisheye-OLSR and the C-OLSR protocols. Simulations show that our protocol outperforms these two solutions.

1 Introduction

Mobile Ad Hoc Networks (MANET) are special communication systems that can be deployed even when no network infrastructure exists or may be installed. They are of great interest for scenarios such as tactical military networks, law enforcement, disaster recovery or sensor networks. In a MANET, nodes are mobile and self-organized to establish multi-hop communications autonomously. Routing in such mobile with resource-constrained (bandwidth and energy) networks is one of

Lucile Canourgues
Rockwell Collins France and IRIT/ENSEEIHT and TeSA lab., 14-16 Port Saint Etienne, F-31000, Toulouse, France, e-mail: lcanourg@rockwellcollins.com

Jerome Lephay and Laurent Soyer
Rockwell Collins France, 6 avenue D. Daurat, BP 20008, F-31701 Blagnac Cedex, France e-mail: jlephay@rockwellcollins.com, lsoyer@rockwellcollins.com

Andre-Luc Beylot
IRIT/ENSEEIHT, 2 rue Camichel, F-31071 Toulouse, France e-mail: andre-luc.beylot@enseeiht.fr

the most challenging issues. In this paper, we will focus on the Optimized Link State Routing (OLSR) [1] protocol, a proactive routing protocol standardized by the MANET WG [2].

OLSR [1] is a well-known link state proactive routing protocol widely employed in deployed MANET systems. Contrary to reactive routing protocols which discover routes on-demand at the time the application needs to communicate through some form of signaling (usually controlled broadcast technique), proactive protocols continually discover and maintain routes to all other nodes in the network. The control overhead associated with the proactive approachs is the main drawback of these solutions. OLSR uses the concept of MultiPoint Relays (MPRs) [3] to achieve an efficient flooding of the control messages in the network and thus to reduce control overhead. Nodes selected as MPRs by neighbor nodes periodically announce this information in their Topology Control (TC) messages. They inform thus the network nodes that they have reachability to the nodes which have selected them as MPR. In route computation, MPRs are used to form the minimal route from a given node to any destination in the network. Therefore, the MPR concept allows OLSR to reduce both the number of broadcast packet transmissions and the size of the link state update packets, leading to an efficient flooding of control message in the network.

Nevertheless, even though OLSR reduces the control overhead in dense and small networks, it still presents scalability issues. Indeed, when the network is scarse, all neighbors are chosen as MPRs and therefore the protocol degrades to a pure link state protocol. Moreover, the overhead of control messages is directly linked to the number of nodes, so it may become unaffordable in large networks with several hundreds of nodes. Finally, since each node must store a route to all other nodes in the network, the associated storage capacity requirement may become too burdensome when the size of the network increases, especially for MANET nodes for which memory resource is limited. These are the reasons why proactive protocols like OLSR are known to have poor scalability properties.

This paper is organized as follows. In section 2, we present the state-of-the-art regarding the solutions proposed to improve OLSR scalability. Then, in section 3, our solution is described. Section 4 provides theoretical and simulation analyses of the overhead generated by our approach compared to the FishEye OLSR and the C-OLSR approaches. Finally, we conclude this article in section 5.

2 Related work

As shown in [4], the control overhead in OLSR is mainly due to TC messages rather than Hello messages used for the neighborhood discovery. Consequently, propositions to improve OLSR scalability focus on reducing the amount of TC messages.

Fisheye-OLSR [5] integrates the fisheye technique into OLSR. The fisheye routing consists in adapting the frequency of the forwarding of topology information to the distance to the source. Thus, nearby nodes receive topology information more frequently than farther nodes. The routing information for a remote destination is

only vague at first but becomes more and more accurate as the message goes closer to the destination. Analytical studies [6] show that this protocol improves the scalability of OLSR. Nevertheless, the problems of storage and overhead that make OLSR poorly scalable are not fully solved. Each node still has to store and compute a route to all potential destination nodes in the network. Moreover, TC messages are still broadcast on the whole network by all the MPRs, even if their frequency is reduced.

To overcome these issues, the solution is to limit each node view of the network by aggregating nodes. Indeed, aggregation enables the reduction of the algorithm complexity and the optimization of the resource (e.g. memory, medium ...) and simplifies the network management. In MANET, the aggregation of nodes is performed thanks to the clustering technique. Clustering allows to introduce levels of hierarchy in the network. Having levels of hierarchy enables the growth of the size of the each node's routing table to be only logarithmic instead of linear with respect to the number of nodes in the network. Based on a clustering protocol, several propositions to enhance the OLSR protocol have been made.

Hierarchical OLSR (HOLSR)[7] proposes a scalable improvement of OLSR based on clustering. Clusterheads are selected based on their higher communications capabilities (data rate, radio range frequency band, battery life ...). Topology information are sent only within the cluster and clusterheads exchange the address of the nodes belonging to their cluster through direct communications. To reach any other destination, data packets are firstly routed through the local clusterhead and then forwarded to the appropriate peer clusterhead. This may lead to suboptimal paths when, for example, the source and the destination are close but belong to different clusters. Moreover, the assumption of the existence of higher capabilities nodes is a strong assumption that may not be verified in tactical MANET.

OLSR tree[8] defines a clustering algorithm to introduce hierarchy in OLSR. The clustering algorithm is based on the connectivity of nodes. The network is divided into trees, where the root of the tree, the clusterhead, is the node having the maximum local connectivity. Once trees are created, a maintenance process is run. A hierarchical routing protocol based on OLSR is then employed. Routing within the tree scope is done with OLSR as if there were no tree. To route to other trees, OLSR is applied on the cluster topology thanks to "super messages" (Super TC, Super Hello, ...) exchanged by clusterheads. When a node needs to send data to a node outside its tree, it first sends the traffic to its root which then forwards the traffic to the destination node following the cluster path. OLSR tree proposed an interesting approach to improve OLSR scalability. Nevertheless, it is dependent on the clustering algorithm which itself is based on connectivity, i.e. a dynamic parameter in mobile networks. Consequently, cluster topology stability may be poor.

The C-OLSR protocol has been presented recently [9] and proposes a modification of OLSR which makes use of clustering to reduce the protocol overhead. Contrary to OLSR Tree, the protocol does not depend on a defined clustering protocol but assumes merely that a clustering algorithm is being executed in the ad hoc network. C-OLSR uses regular OLSR inside every cluster and TC messages forwarding is thus limited within the scope of a cluster. Then, the authors choose

to leverage the same mechanisms of plain OLSR to the level of clusters. Therefore, they define new C-Hello and C-TC messages to emulate the behavior of an OLSR node by a cluster. C-MPR clusters are elected thanks to the C-Hello messages. The C-Hello messages must be forwarded over the entire neighbor clusters so that each node of a cluster may compute its own C-MPR set and the C-TC messages must also be forwarded over the entire clusters that are selected as C-MPR clusters.

As in OLSR Tree, applying OLSR at the cluster level imposes to exchange some sort of Hello (Super-Hello or C-Hello) and TC (Super-TC or C-TC) messages which may generate an important overhead. Moreover, in case of loss of one of these messages, the integrity of the routing function may be jeopardized. We propose a solution to adapt OLSR to a clusterhead environment where regular OLSR is applied inside every cluster for intra-cluster communications but where inter-cluster communications do not rely on a version of OLSR at the cluster level contrary to what is done in both OLSR Tree and C-OLSR.

3 Protocol Description

3.1 Overview

We propose a routing protocol based on OLSR which aims at improving the scalability features of the OLSR protocol in large-scale ad hoc networks. Our protocol makes use of clustering to greatly reduce the topology overhead and the routing table size. The routing protocol is fully independent from the clustering protocol used. The propagation of the topology control information is limited within the cluster. Each node annonces in its Hello messages the address of its clusterhead. For this purpose, we define a new Link Code. Contrary to other OLSR-based approaches which use clustering such as OLSR-Tree, OLSR is not applied on the clusterhead topology for out-of-cluster routing purposes. Indeed, rather than applying the complex OLSR message exchange and MPR selection on clusterheads, in our solution clusterheads only send special TC_Cluster messages over the network. Thanks to these TC_messages, each node knows the next hop node towards the clusterhead the destination depends on. We assume that a clustering protocol is employed within the ad hoc network. A K-clustering algorithm creating clusters with diameter larger than 2 hops is recommended. We also assume that every node is aware of its clusterhead address.

3.2 Hello messages

The first modification we performed on the OLSR protocol was that each node must include its clusterhead membership information, i.e. the address of its clusterhead,

in its Hello message. Hello message format and link code format are defined in the RFC 3626.

With the aim to be compliant with the regular OLSR protocol (i.e. we want that our protocol can be used in networks where both the "regular" OLSR and our protocol co-exist), the Hello message format is not modified. A new Link Code is proposed. Consequently, in a Hello message advertising a clusterhead address, one of the link code blocks will be dedicated to the clusterhead address of the node sending the Hello. The new link code is as follows: the Neighbor Type is set to a new CH_NEIGH value and the Link Type field is set to the UNSPEC_Link value. The Neighbor Interface Address list is composed of one address, the clusterhead address. When a non-OLSR-cluster node receives such an Hello message, it discards the clusterhead related part since it does not understand the link code but the other blocks of the message advertising the neighborhood status are processed as usual.

Upon reception of a Hello message with a clusterhead address, the clusterhead address is saved in the neighbor set of the sending node. Therefore, a new field, the clusterhead_address field, has been added to the neighbor tuple which was previously made of three fields.

3.3 TC messages

A regular version of the OLSR protocol is used within each cluster. TC messages are never forwarded by a node that does not belong to the same cluster as the originator of the TC message. That way, TC message propagation is restricted to the cluster area. The MPR selection algorithm is performed without any consideration of the cluster for network consistency purposes. Therefore, each MPR sends periodic TC messages containing the list of its MPR selectors, i.e. the nodes which select it as MPR. When receiving a TC message, a node processes it following the algorithm described in RFC 3626 [1]. The forwarding decision is then based on the clusterhead of the sender, i.e. the node that has just forwarded the message (and not the originator). The TC forwarding algorithm is roughly the same as the default forwarding algorithm described in the [1] except the first step as illustated below.

> If the sender interface address of the message is not detected to be in the symmetric 1-hop neighborhood of the node, or **if the clusterhead address corresponding to the sender interface address is not the same as our clusterhead address**, the forwarding algorithm MUST silently stop here (and the message MUST NOT be forwarded).

There is no need to add the clusterhead address in the TC messages. Indeed, a node is able to know which is the clusterhead of the node having forwarded the message thanks to its neighbor set. Moreover, the clusterhead of the node that has forwarded the message is necessarily the same as the one that has previously for-

warded the message to it, otherwise the message would not have been forwarded. Step by step, the clusterhead of the TC message originator is the same as the clusterhead of the sending node of a received TC message.

3.4 TC_Cluster Message

From the protocol described previously, a node is able to compute route to all the nodes in its cluster. Border nodes[1] are also able to compute routes to nodes belonging to neighbor clusters since they receive (but not forward) TC messages generated within these clusters. For the routes to nodes belonging to different clusters, our

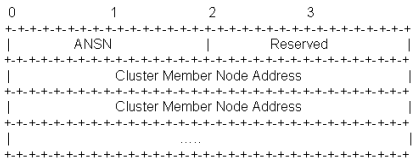

Fig. 1 TC_Cluster message format

approach is that each node should know the next hop toward the clusterhead the destination depends on. Then, once the data packet arrives in the cluster of the destination node, the intermediate node knows the exact route to the destination. To achieve such behavior, cluster topology information must be sent over the network. OLSR-Tree approach is to reproduce the OLSR protocol at the cluster level to create some "cluster paths". The approach we follow is different. We define a new TC_Cluster message that is sent by clusterheads over the network using the MPR flooding algorithm. This message does not contain the list of the MPR selectors of the clusterhead but rather the addresses of nodes belonging to its cluster. Since this message is flooded on the overall network, each node can maintain a node/cluster membership table and can therefore determine to which cluster a destination node belongs to. Nevertheless, knowing the clusterhead the destination node is related to is not enough to route a packet towards this destination node. Indeed, the path to the clusterhead or at least the next hop node on the path to the clusterhead is needed. This next hop information is retrieved when receiving the TC_Cluster message. Indeed, when a node receives a TC_Cluster message, it registers as its next hop to the clusterhead sending the message the node that has just forwarded the message, assuming that this is the first time this message is received. Since the message is flooded over the network, a node may receive several copies of a TC_Cluster message. Nevertheless, the first copy received is the only one considered for the next hop information since it has necessarily taken the faster, less congested path. The other copies are discarded.

[1] A border router is a node that is one hop away from a node that belongs to a different cluster i.e. a node that has a different clusterhead.

Figure 1 gives the format of the TC_Cluster message. The originator of a TC_Cluster message is the clusterhead, and the source address in the IP header is the candidate next-hop node toward the clusterhead. The number of hops to reach the clusterhead can also be computed through the TC_Cluster message thanks to the following formula: $TTL\_TC\_Cluster - TTL\_of\_the\_received\_message$, where TTL_TC_Cluster is a constant and is the TTL value the originator of a TC_Cluster message must set in the TTL field of the message and TTL_of_the_received_message is the TTL value indicated in the TTL field of the received TC_Cluster message. The TC_Cluster periodicity is lower than the TC periodicity assuming that the clustering protocol creates stable clusters.

3.5 Sending and Forwarding Data Packets

When a node has a data packet to send:

- if it knows the destination from its routing table (i.e. the node belongs to its cluster or the destination is a clusterhead), it sends the packet to the next hop indicated in its routing table
- if the destination is not in its routing table, it performs a look in the node/cluster membership table to know which cluster the node belongs to.
  - If the destination is not in the table, the packet is discarded
  - If the destination is in the table, the node looks into its routing table for the next hop to the clusterhead. The destination address of the node is not changed. When the next hop receives the data, the same process is performed.

4 Theoretical and Simulation Performance Analysis

In this section we want to evaluate the overhead generated by our protocol. Since our objective is mainly to reduce the control overhead caused by TC messages, we will only consider the TC messages control overhead. Firstly, we give theoretical analyses of the control overhead of our protocol and of the Fisheye-OLSR protocol in order to verify that employing a clustering approach allows to improve the overhead compared to the Fisheye technique. Then, we compare our approach to the C-OLSR and the Fisheye-OLSR protocols through a simulation study. In this simulation study, the MAC layer is considered as a perfect MAC since we are mainly interested in this paper in the evaluation of the control overhead of our protocol with respect to other OLSR scalable protocols. In future performance evaluation works, we will consider end-to-end performance such as the packet delivery ratio or the delay by implementing the protocol in a discrete event simulator that integrates a more realistic MAC layer and mobility such as OPNET or NS2.

4.1 Theoretical Analyis

4.1.1 Nework Model and Parameters

The network is represented by a Poisson Point Process over the plan denoted S with intensity λ. Let N be the number of nodes in the network. N follows a Poisson law with intensity $\lambda * S$. λ represents the mean number of nodes per unity of surface. It follows that the density of the network $M = \lambda$, which means that on average each node has M neighbors or that on a unit disk centered on a node, there are on average M nodes. Therefore, the number of nodes in the K-hop neighborhood of a node is equal to the number of nodes in a disk a radius K which is on average K^2M. Moreover, the radius of the network is $\sqrt{N/M}$. Let M_R be the average number of MPRs selected by a node with a neighborhood size M. It has been shown in [10] and [6] that $M_R \leq (9\pi^2 M)^{1/3}$ and that $M_R \sim \beta M^{1/3}$ when $M \to \infty$ with $\beta \approx 5$.

The number of retransmissions of a TC message in the K-Hop neighborhood is equal to the number of MPRs in the K-Hop neighborhood, which is on average equal to the number of nodes in the K-Hop neighborhood times the probability for a node to be an MPR. Consequently, the number of retransmissions of a TC message in the K-Hop neighborhood of a node is on average : $M_R/M * K^2M = M_RK^2$. Then it follows that the number of nodes at exactly K hops of a node that may retransmit a TC message is on average : $M_R/M * (K^2 - (K-1)^2)M = M_R(K^2 - (K-1)^2)$.

4.1.2 Fisheye OLSR

In the Fisheye OLSR improvement, the period of the TC messages received from a node increases with the distance to the sending node. We can define a function F that gives the period of the TC messages based on the number of hops from the source, i.e. the TTL set in the messages. Let us consider the function $F : F(x) = \frac{4x}{3+x}$ as in [6] where x represents the TTL and $F(x)$ is the period of the TC message for the TTL x. The overhead generated by a TC message sent by an MPR in bits/s is :

$$\sum_{K=2}^{\sqrt{N/M}} \frac{1}{F(K)}(((K-1)^2 - (K-2)^2)M_R + 1) * TC_{size} \tag{1}$$

It should be noted that for a TTL of k, the message will be retransmitted $(k-1)$ times. TC_{size} is the size of a TC message in bits and is on average $(M_R + 5) * 8$ bits. Finally it follows that the overhead in bits/s due to the TC messages in the Fisheye OLSR protocol is on average:

$$(1) * \text{number of MPR in the network} = (1) * M_R N/M \tag{2}$$

4.1.3 Our approach

Let $TC_{Interval}$ be the period of the TC message. The default value is 5 seconds. Let C be the mean number of clusters in the network. It has be shown that for the Max-Min heuristic [11], an upper bound of the mean number of clusters can be found :

$$\mathbf{E}\left[\text{Clusterhead \# in S}\right] \leq \lambda \cdot \nu(S) \cdot \left(1 + \sum_{n=1}^{\infty} \frac{1}{n} \frac{E^n}{n!}\right) exp\left(-E\right)$$

with $E = \lambda \pi R^2$, where R is the propagation range of a node and $\nu(S)$ is the Lebesgue measure of S. This upper bound is computed for a radius of 1. It is shown that for radius greater than 1, the set of clusterheads is included in the one computed for radius 1. Therefore, this upper bound becomes less and less accurate as the radius increases.

In our approach, we have to distinguish the overhead generated by TC messages forwarding within each cluster from the overhead generated by the forwarding of the TC_Cluster messages. An upper bound of the mean number of nodes per cluster is equal to r^2M nodes, where r is the radius of the cluster. The overhead due to the forwarding of a TC message sent by an MPR within a cluster in bits/s is thus bounded by the following upper bound:

$$\left(1 + r^2 M_R\right) * TC_{size}/TC_{Interval} \tag{3}$$

Moreover, the mean overhead generated by the forwarding of a TC_Cluster message sent by a clusterhead in bits/s is :

$$(1 + NM_R/M) * TC\_Cluster_{size}/TC\_Cluster_{Interval} \tag{4}$$

Finally, the control overhead due to the TC and TC_Cluster message forwarding is bounded:

$$\mathbf{E}\left[\text{control message overhead}\right] \leq (4) * (NM_R/M) + (3) * C \tag{5}$$

4.1.4 Comparison of the theoretical bounds of the control overhead

In this section, we compare the theoretical overhead of the Fisheye OLSR and our proposal based on the expressions given in the previous sections. One should note that in the following results, the overhead due to the clustering algorithm has been added to the TC and TC_Cluster control overhead for our solution. As illustrated by figure 2, when the number of nodes increases, the overhead due to the TC messages with the Fisheye solution greatly increases whereas it increases slowly with our proposition. When the number of nodes is low (under 200 nodes), the overheads of the two approaches are close. This underlines a potential limit on the network size for the use of a clustering algorithm. Figure 3 presents the overhead of the TC messages in a network of 500 nodes as the density increases. It shows that the per-

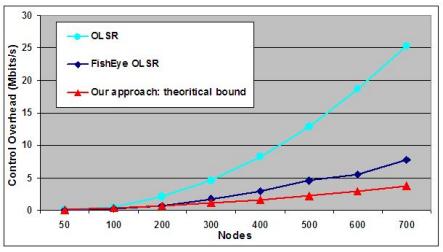

Fig. 2 TC overhead comparison between Fisheye OLSR and our protocol versus the number of nodes

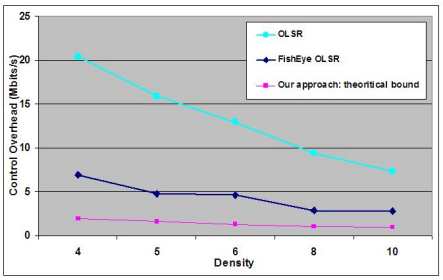

Fig. 3 TC overhead comparison between Fisheye OLSR and our protocol versus the density

formance of Fisheye OLSR improves as density increases just as the OLSR protocol itself does. This result is due to the fact that when density increases, the diameter of the network decreases, and the MPR algorithm is more efficient. Therefore, fewer retransmissions are needed to send TC messages to the network. However, our protocol still outperforms the Fisheye approach even in very dense networks. With our approach, the density has less impact on the control overhead.

4.2 Performance evaluation based on simulation

In this section, we compare the overhead of different approaches proposed to improve the scalability of OLSR. We will consider the fisheye OLSR solution and the C-OLSR solution to which we will compare our approach. These three protocols have been implemented thanks to the Scilab 4.1.2 [12] simulation tool. For the clustering, we implement the generalized max-min clustering algorithm [11]. Since we are mainly interested in the control overhead, we compare the control overhead caused by either the TC messages or their substitutes in each of these protocols:

- TC messages for the Fisheye-OLSR protocol
- TC messages forwarded within each cluster, C-Hello messages, C-TC messages and control message overhead due to the clustering protocol for the C-OLSR protocol
- TC messages forwarded within each cluster, TC_Cluster messages, and control message overhead due to the clustering protocol for our protocol.

Figure 4 presents a comparison between the upper theoretical bound and the simulated values of the overhead of the Fisheye OLSR and our solution. This results prove that the theoretical expressions really give upper bounds but it also shows that these bounds are rather far from the simulation values. Moreover, if the theoretical bound of our approach is higher than the theoretical bound of Fisheye-OLSR for small networks, the simulation shows that even with small networks, our approach performs better than Fisheye OLSR. Figure 5 presents a comparison of the control

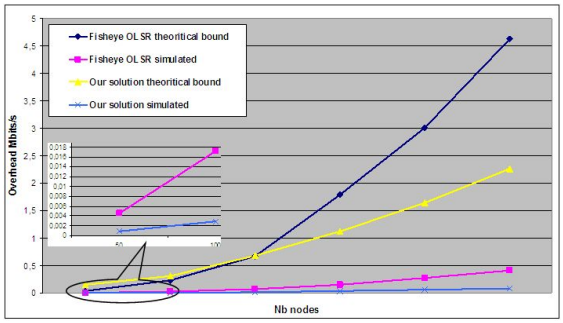

Fig. 4 Comparison of the theoretical and simulated values of the control overhead

overhead of Fisheye OLSR, C-OLSR and our solution as a function of the number of nodes. The results prove that employing a clustering algorithm allows to greatly improve the scalability of OLSR compared to the Fisheye solution. Moreover, we show that our solution presents better scalability compliance than the C-OLSR solution where OLSR is applied on top of the cluster topology. The difference between the overhead of C-OLSR and our approach results only from the inter-cluster communications approach since both the clustering and the intra-cluster TC forwarding is similar in these two solutions. Therefore, the results prove that a solution that does not re-use the OLSR algorithm on top of the cluster topology presents better efficiency in term of control overhead than applying an adapted version of OLSR on the cluster topology.

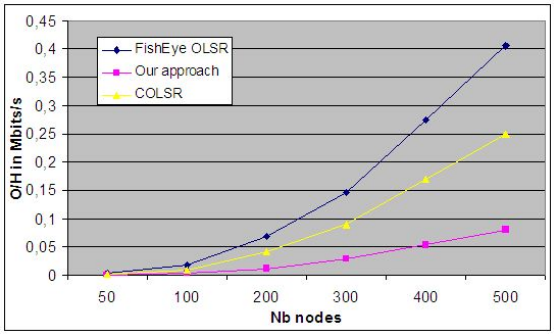

Fig. 5 Comparison of the control overhead of Fisheye OLSR, C-OLSR and our approach

5 Conclusion

This paper presents a scalable routing protocol for Mobile Ad hoc Networks that is based on and improves the well known proactive unicast routing protocol OLSR to make it scalable. The protocol assumes that nodes are gathered in clusters thanks to a clustering algorithm. The regular OLSR protocol is applied within the clusters and a new message type is defined to allow inter-cluster routing. These new messages, called TC_Cluster, are sent by the clusterheads and contain the list of the nodes belonging to their cluster. TC_Cluster messages are broadcast over the entire network thanks to the optimized MPR flooding. Theoretical and simulation analyses of the control overhead of our protocol compared to the Fisheye-OLSR and the C-OLSR show that our approach significantly reduces the control overhead as the number of nodes in the network increases.

Acknowledgements This work has been carried out under the financial support of Rockwell Collins and under the co-funding of the French research ministry and the European Social Fund.

References

1. T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)." RFC 2636, October 2003.
2. J. Maker and I. Chakeres, "Mobile Ad-hoc Networks (Manet)." Charter, [on line] http://www.ietf.org/html.charters/manet-charter.html.
3. A.Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying: An Efficient Technique for flooding in Mobile Wireless Networks." RR-3898, INRIA, Feb 2000.
4. A. Laouti, P. Mhlethaler, A. Najid, and E. Plakoo, "Simulation Results of the OLSR Routing Protocol for Wireless Network," in *Med-Hoc-Net workshop*, (Sardegna, Italy), 2002.
5. T. Clausen, "Combining Temporal and Spatial Partial Topology for MANET routing - Merging OLSR and FSR," in *IEEE WPMC'03*, (Yokosuka, Japan), 2003.
6. C. Adjih, E. Baccelli, T. Clausen, P. Jacquet, and G. Rodolakis, "Fish eye OLSR scaling properties," *Journal of communication and networks (JCN)*, vol. 6, no. 4, pp. 343–351, 2004.
7. Y. Ge, L. Lamont, and L. Villasenor, " Hierarchical OLSR - A Scalable Proactive Routing Protocol for Heterogeneous Ad Hoc Networks," in *WiMob'05*, Montreal, Canada 2005.
8. E. Baccelli, "OLSR Scaling with Hierarchical Routing and Dynamic Tree Clustering," in *IASTED International Conference on Networks and Communication Systems (NCS)*, (Chiang Mai, Thailand), March 2006.
9. F. Ros and P. Ruiz, "Cluster-based OLSR Extension to Reduce Control Overhead in Mobile Ad Hoc Networks," in *IWCMC'07*, August 2007.
10. P. Jacquet, A. Laouiti, P.Minet, and L. Viennot, "Performance evaluation of multipoint relaying in mobile ad hoc networks," in *Networking 2002*, (Pisa), 2002.
11. A. D. de Clauzade, M. Marot, and M. Becker, "An analysis of the generalised Max-Min d-cluster formation heuristic," in *Med-Hoc-Net 2007 Workshop*, (Ionian University, Corfu, Greece), June 2007.
12. "Scilab, plateforme open source de calcul scientifique." http://www.scilab.org.